ISIS-II PL/M-86 V2.0 COMPILATION OF MODULE SUB
OBJECT MODULE PLACED IN SUBMIT.OBJ
COMPILER INVOKED BY: :F0: SUBMIT.PLM DATE(MARCH 16, 1982) XREF OPTIMIZE(3) DEBUG

```
 1          sub:
            do;
            /* modified 7/26/79 to work with cpm 2.0, modu : number not zero */
            /* modified 10/5/80 for 8086                  */


 2  1       mon1: procedure(f,a) external;
 3  2           declare f byte, a address;
                /* bdos interface, no returned value */
 4  2           end mon1;

 5  1       mon2: procedure(f,a) byte external;
 6  2           declare f byte, a address;
                /* bdos interface, return byte value */
 7  2           end mon2;

 8  1       declare
               copyright(*) byte data
                  (' copyright(c) 1977, digital research ');

 9  1       declare
               ln(5) byte initial('001 $'),
               ln1 byte at(.ln(0)),
               ln2 byte at(.ln(1)),
               ln3 byte at(.ln(2)),
               dfcb(33) byte initial(1,'$$$     SUB',0,0,0),
               drec byte at(.dfcb(32)),  /* current record */
               buff(128) byte external;   /* default buffer */
               sfcb(33)  byte external;   /* default fcb */

10  1       submit: procedure public;

               /* t h e   c p / m   ' s u b m i t '   f u n c t i o n

                  copyright (c) 1976, 1977, 1978, 1980
                  digital research
                  box 579
                  pacific grove, ca.
                  93950
               */
11  2       declare lit literally 'literally',
               dcl lit 'declare',
               proc lit 'procedure',
               addr lit 'address',
               ctll lit '0ch',
               lca  lit '110$0001b',  /* lower case a */
               lcz  lit '111$1010b',  /* lower case z */
               endfile lit '1ah';    /* cp/m end of file */

12  2       declare
               true literally '1',
```

CP/M-86 v.1.1 (Vol. II)
COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. # C81-162

```
                false literally '0',
                forever literally 'while true',
                cr literally '13',
                lf literally '10',
                what literally '63';

 13  2      print: procedure(a);
 14  3          declare a address;
                /* print the string starting at address a until the
                next dollar sign is encountered */
 15  3          call mon1(9,a);
 16  3          end print;


 17  2      declare dcnt byte;

 18  2      open: procedure(fcb);
 19  3          declare fcb address;
 20  3          dcnt = mon2(15,fcb);
 21  3          end open;

 22  2      close: procedure(fcb);
 23  3          declare fcb address;
 24  3          dcnt = mon2(16,fcb);
 25  3          end close;

 26  2      delete: procedure(fcb);
 27  3          declare fcb address;
 28  3          call mon1(19,fcb);
 29  3          end delete;

 30  2      diskread: procedure(fcb) byte;
 31  3          declare fcb address;
 32  3          return mon2(20,fcb);
 33  3          end diskread;

 34  2      diskwrite: procedure(fcb) byte;
 35  3          declare fcb address;
 36  3          return mon2(21,fcb);
 37  3          end diskwrite;

 38  2      make: procedure(fcb);
 39  3          declare fcb address;
 40  3          dcnt = mon2(22,fcb);
 41  3          end make;

 42  2      move: procedure(s,d,n);
 43  3          declare (s,d) address, n byte;
 44  3          declare a based s byte, b based d byte;
 45  3              do while (n := n - 1) <> 255;
 46  4              b = a; s = s + 1; d = d + 1;
 49  4              end;
 50  3          end move;

 51  2      declare oldsp address;   /* calling program's stack pointer */

 52  2      error: procedure(a);
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
53  3       declare a address;
54  3       call print(.(cr,lf,'$'));
55  3       call print(.('Error On Line $'));
56  3       call print(.ln1);
57  3       call print(a);
58  3       call mon1(0,0);
             /* return to ccp */
59  3       end error;

60  2   declare sstring(128) byte, /* substitute string */
         sbp byte;            /* source buffer pointer (0-128) */


61  2   setup: procedure;
         /* move buffer to substitute string */
62  3       call move(.buff(1),.sstring(0),127);
63  3       sstring(buff(0))=0; /* mark end of string */
64  3       call move(.('SUB'),.sfcb(9),3); /* set file type to sub */
65  3       call open(.sfcb(0));
66  3       if dcnt = 255 then
67  3           call error(.('No ''SUB'' File Present$'));
         /* otherwise file is open - read subsequent data */
68  3       sbp = 128; /* causes read below */

69  3       end setup;


70  2   getsource: procedure byte;
         /* read the next source character */
71  3       declare b byte;
72  3       if sbp > 127 then
73  3           do; if diskread(.sfcb(0)) <> 0 then
75  4               return endfile;
76  4           sbp = 0;
77  4           end;
78  3       if (b := buff((sbp:=sbp+1)-1)) = cr then
79  3           do; /* increment line */
80  4           if (ln3 := ln3 + 1) > '9' then
81  4               do; ln3 = '0';
83  5               if (ln2 := ln2 + 1) > '9' then
84  5                   do; ln2 = '0';
86  6                   ln1 = ln1 + 1;
87  6                   end;
88  5               end;
89  4           end;
         /* translate to upper case */
90  3       if (b-61h) < 26 then /* lower case alpha */
91  3           b = b and 5fh; /* change to upper case */
92  3       return b;
93  3       end getsource;

94  2   writebuff: procedure;
         /* write the contents of the buffer to disk */
95  3       if diskwrite(.dfcb) <> 0 then /* error */
96  3           call error(.('Disk Write Error$'));
97  3       end writebuff;
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
 98  2      declare rbuff(2048) byte, /* jcl buffer */
                rbp address,      /* jcl buffer pointer */
                rlen byte;     /* length of current command */

 99  2      fillrbuff: procedure;
100  3          declare (s,ssbp) byte;  /* sub string buffer pointer */

101  3          notend: procedure byte;
                    /* look at next character in sstring, return
                    true if not at the end of the string - char passed
                    back in 's' */
102  4              if not ((s := sstring(ssbp)) = ' ' or s = 0) then
103  4                  do;
104  5                  ssbp = ssbp + 1;
105  5                  return true;
106  5                  end;
107  4              return false;
108  4              end notend;

109  3          deblankparm: procedure;
                    /* clear to next non blank substitute string */
110  4                  do while sstring(ssbp) = ' ';
111  5                  ssbp = ssbp + 1;
112  5                  end;
113  4              end deblankparm;

114  3          putrbuff: procedure(b);
115  4             declare b byte;
116  4              if (rbp := rbp + 1) > last(rbuff) then
117  4                  call error(.('Command Buffer Overflow$'));
118  4              rbuff(rbp) = b;
                    /* len: c1 ... c125 :00:$ = 128 chars */
119  4              if (rlen := rlen + 1) > 125 then
120  4                  call error(.('Command Too Long$'));
121  4              end putrbuff;

122  3          declare (reading,b) byte;
                /* fill the jcl buffer */
123  3          rbuff(0),rbp = 0;
124  3          reading = true;
                    do while reading;
126  4              rlen = 0; /* reset command length */
127  4                  do while (b:=getsource) <> endfile and b <> cr;
128  5                  if b <> lf then
129  5                      do; if b = '$' then /* copy substitute string */
131  6                         do; if (b:=getsource) = '$' then
                                  /* $$ replaced by $ */
133  7                              call putrbuff(b); else
134  7                          if (b := b - '0') > 9 then
135  7                              call error(.('Parameter Error$')); else
136  7                              do; /* find string 'b' in sstring */
137  8                              ssbp = 0; call deblankparm; /* ready to scan sstring */
139  8                                  do while b <> 0; b = b - 1;
                                        /* clear next parameter */
141  9                                      do while notend;
142 10                                      end;
143  9                                  call deblankparm;
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
144  7                          end;
                              /* ready to copy substitute string from position ssbp */
145  8                          do while notend;
146  9                          call putrbuff(s);
147  9                          end;
148  8                        end;
149  7                     end; else /* not a '$' */
150  6                   if b = '^' then /* control character */
151  6                      do; /* must be ^a ... ^z */
152  7                      if (b:=getsource - 'A') > 25 then
153  7                          call error(.('Invalid Control Character$'));
                            else
154  7                           call putrbuff(b+1);
155  7                      end; else /* not $ or ^ */
156  6                   call putrbuff(b);
157  6                   end;
158  5                end; /* of line or input file - compute length */
159  4             reading = b = cr;
160  4             call putrbuff(rlen); /* store length */
161  4             end;
                 /* entire file has been read and processed */
162  3          end fillrbuff;

163  2      makefile: procedure;
             /* write resulting command file */
164  3          declare i byte;
165  3          getrbuff: procedure byte;
166  4              return rbuff(rbp := rbp - 1);
167  4              end getrbuff;

168  3          call delete(.dfcb);
169  3          drec = 0; /* zero the next record to write */
170  3          call make(.dfcb);
171  3          if dcnt = 255 then call error(.('Directory Full$'));
173  3              do while (i := getrbuff) <> 0;
                   /* copy i characters to buffer */
                   /* 00 $ at end of line gives 1.3 & 1.4 compatibility */
174  4              buff(0) = i; buff(i+1) = 00; buff(i+2) = '$';
177  4                  do while i > 0;
178  5                  buff(i) = getrbuff; i=i-1;
180  5                  end;
                   /* buffer filled to $ */
181  4              call writebuff;
182  4              end;
183  3          call close(.dfcb);
184  3          if dcnt = 255 then call error(.('Cannot Close, Read/Only?$'));
186  3          end makefile;

               /* enter here from the ccp with the fcb set */

187  2          call setup;
188  2          call fillrbuff;
189  2          call makefile;
190  2          end submit;
191  1      end;
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES
44	0000H	1	A. . . . . BYTE BASED(S) 46
13	0004H	2	A. . . . . WORD PARAMETER AUTOMATIC 14 15
52	0004H	2	A. . . . . WORD PARAMETER AUTOMATIC 53 57
5	0000H	2	A. . . . . WORD PARAMETER 6
2	0000H	2	A. . . . . WORD PARAMETER 3
11			ADDR . . . . . LITERALLY
71	00ACH	1	B. . . . . BYTE 78 90 91 92
44	0000H	1	B. . . . . BYTE BASED(D) 46
114	0004H	1	B. . . . . BYTE PARAMETER AUTOMATIC 115 118
122	08B1H	1	B. . . . . BYTE 127 128 130 132 133 134 139 140 150 152 154 156 159
9	0000H	128	BUFF . . . . . BYTE ARRAY(128) EXTERNAL(2) 62 63 78 174 175 176 178
22	0031H	19	CLOSE. . . . . PROCEDURE STACK=000AH 183
8	0000H	37	COPYRIGHT. . . . . BYTE ARRAY(37) DATA
12			CR . . . . . LITERALLY 54 78 127 159
11			CTLL . . . . . LITERALLY
42	0006H	2	D. . . . . WORD PARAMETER AUTOMATIC 43 44 46 48
11			DCL. . . . . LITERALLY
17	002AH	1	DCNT . . . . . BYTE 20 24 40 66 171 184
109	02BEH	24	DEBLANKPARM. . . . PROCEDURE STACK=0002H 138 143
26	0044H	16	DELETE . . . . . PROCEDURE STACK=000AH 168
9	0009H	33	DFCB . . . . . BYTE ARRAY(33) INITIAL 9 95 168 170 183
30	0054H	16	DISKREAD . . . . . PROCEDURE BYTE STACK=000AH 74
34	0064H	16	DISKWRITE. . . . . PROCEDURE BYTE STACK=000AH 95
9	0027H	1	DREC . . . . . BYTE AT 169
11			ENDFILE. . . . . LITERALLY 75 127
52	00ACH	43	ERROR. . . . . PROCEDURE STACK=0010H 67 96 117 120 135 153 172 185
5	0000H	1	F. . . . . BYTE PARAMETER 6
2	0000H	1	F. . . . . BYTE PARAMETER 3
12			FALSE. . . . . LITERALLY 107
30	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 31 32
22	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 23 24
18	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 19 20
38	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 39 40
34	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 35 36
26	0004H	2	FCB. . . . . WORD PARAMETER AUTOMATIC 27 28
99	01A4H	227	FILLRBUFF. . . . . PROCEDURE STACK=001AH 188
12			FOREVER. . . . . LITERALLY
165	0386H	18	GETRBUFF . . . . . PROCEDURE BYTE STACK=0002H 173 178
70	011DH	112	GETSOURCE. . . . . PROCEDURE BYTE STACK=000EH 127 132 152
164	08B2H	1	I. . . . . BYTE 173 174 175 176 177 178 179
			LAST . . . . . BUILTIN 116
11			LCA. . . . . LITERALLY
11			LCZ. . . . . LITERALLY
12			LF . . . . . LITERALLY 54 128
11			LIT. . . . . LITERALLY 11
9	0004H	5	LN . . . . . BYTE ARRAY(5) INITIAL 9
9	0004H	1	LN1. . . . . BYTE AT 56 86
9	0005H	1	LN2. . . . . BYTE AT 83 85

```
  9  0006H    1  LN3. . . . . .   BYTE AT         80   82
 38  0074H   19  MAKE . . . . . .  PROCEDURE STACK=000AH          170
163  030EH  120  MAKEFILE . . . . .  PROCEDURE STACK=0018H        139
  2  0000H      MON1 . . . . . .  PROCEDURE EXTERNAL(0) STACK=0000H          15   28   58
  5  0000H      MON2 . . . . . .  PROCEDURE BYTE EXTERNAL(1) STACK=0000H        20   24   32   36   40
 42  0087H   37  MOVE . . . . . .  PROCEDURE STACK=0008H          62   64
 42  0004H    1  N. . . . . .     BYTE PARAMETER AUTOMATIC       43   45
101  0287H   55  NOTEND . . . . .  PROCEDURE BYTE STACK=0004H        141  145
 51  0000H    2  OLDSP. . . . . .  WORD
 18  001EH   19  OPEN . . . . . .  PROCEDURE STACK=000AH          65
 13  000EH   16  PRINT. . . . . .  PROCEDURE STACK=000AH          54   55   56   57
 11              PROC . . . . . .  LITERALLY
114  02D6H   56  PUTRBUFF . . . . .  PROCEDURE STACK=0018H       133  146  154  156  160
 98  0002H    2  RBP. . . . . .   WORD       116  118  123  166
 98  00ADH 2048  RBUFF. . . . . .  BYTE ARRAY(2048)        116  118  123  166
122  08B0H    1  READING. . . . .  BYTE       124  125  159
 98  08ADH    1  RLEN . . . . . .  BYTE       119  126  160
100  08AEH    1  S. . . . . .     BYTE       102  146
 42  0003H    2  S. . . . . .     WORD PARAMETER AUTOMATIC       43   44   46   47
 60  00ABH    1  SBP. . . . . .   BYTE        68   72   76   78
 61  00D7H   70  SETUP. . . . . .  PROCEDURE STACK=0014H         187
  9  0000H   33  SFCB . . . . . .  BYTE ARRAY(33) EXTERNAL(3)          64   65   74
100  08AFH    1  SSBP . . . . . .  BYTE       102  104  110  111  137
 60  002BH  128  SSTRING. . . . .  BYTE ARRAY(128)       62   63  102  110
  1  0000H      SUB. . . . . .    PROCEDURE STACK=0000H
 10  0000H   14  SUBMIT . . . . .  PROCEDURE PUBLIC STACK=001EH
 12              TRUE . . . . . .  LITERALLY       105  124
 12              WHAT . . . . . .  LITERALLY
 94  013DH   23  WRITEBUFF. . . .  PROCEDURE STACK=0014H         181
```

MODULE INFORMATION:

```
CODE AREA SIZE     = 0398H    920D
CONSTANT AREA SIZE = 00DCH    220D
VARIABLE AREA SIZE = 08B3H   2227D
MAXIMUM STACK SIZE = 001EH     30D
277 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-86 COMPILATION
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____